

Hit List

[First Hit](#)[Clear](#)[Generate Collection](#)[Print](#)[Fwd Refs](#)[Bkwd Refs](#)[Generate OACS](#)

Search Results - Record(s) 1 through 1 of 1 returned.

☐ 1. Document ID: US 7155681 B2

L18: Entry 1 of 1

File: USPT

Dec 26, 2006

US-PAT-NO: 7155681

DOCUMENT-IDENTIFIER: US 7155681 B2

TITLE: Platform-independent distributed user interface server architecture

DATE-ISSUED: December 26, 2006

PRIOR-PUBLICATION:

DOC-ID

DATE

US 20020109718 A1

August 15, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Mansour; Peter M	Kirkland	WA		US
Schwitters; Chad Arthur	Redmond	WA		US

US-CL-CURRENT: [715/762](#); [715/744](#), [715/752](#), [715/763](#), [715/764](#), [715/765](#)

Full	Title	Citation	Front	Review	Classification	Date	Reference	Abstracts	References	Claims	KWIC	Draw D
------	-------	----------	-------	--------	----------------	------	-----------	-----------	------------	--------	------	--------

[Clear](#)[Generate Collection](#)[Print](#)[Fwd Refs](#)[Bkwd Refs](#)[Generate OACS](#)

Terms

Documents

L17 and @pd > 20060919

1

Display Format:

[Change Format](#)[Previous Page](#)[Next Page](#)[Go to Doc#](#)

[First Hit](#) [Fwd Refs](#)[Previous Doc](#)[Next Doc](#)[Go to Doc#](#)**End of Result Set**

Generate Collection

Print

L18: Entry 1 of 1

File: USPT

Dec 26, 2006

DOCUMENT-IDENTIFIER: US 7155681 B2

TITLE: Platform-independent distributed user interface server architecture

PRIOR-PUBLICATION:

DOC-ID

DATE

US 20020109718 A1

August 15, 2002

Abstract Text (1):

A distributed user interface (UI) system includes a client device configured to render a UI for a server-based application. The client device communicates with a UI server over a network such as the Internet. The UI server performs formatting for the UI, which preferably utilizes a number of native UI controls that are available locally at the client device. In this manner, the client device need only be responsible for the actual rendering of the UI. The source data items are downloaded from the UI server to the client device when necessary, and the client device populates the UI with the downloaded source data items. The client device employs a cache to store the source data items locally for easy retrieval.

DATE ISSUED (1):

20061226

Brief Summary Text (6):

The number of users receiving data services via the Internet and wireless data networks continues to grow at a rapid pace. For example, millions of people have traditional access to the Internet and many people use web-capable wireless telephones. In addition, a growing number of people own handheld computers or personal digital assistants (PDAs), many of which are capable of establishing a traditional and/or a wireless connection to the Internet.

Brief Summary Text (7):

At the heart of this technological explosion are the data-capable Internet appliances. These devices encompass a wide range of form factors: web-enabled telephones, smart telephones, PDAs, handheld gaming machines, and other devices. By nature these devices are small, portable, affordable, and offer instant access to valuable data such as personal information manager (PIM) data and email, as well as entertainment such as gaming, music, and streaming video. The combination of a handheld computing device (HCD) and a wireless network connection is extremely intriguing to the end user, offering a substantially higher value proposition than the HCD has ever held before. With this change, longtime benefits such as portability, instant power-up, and long battery life become much more valuable. The appeal of constant connectivity without the inconvenience of carrying and waiting for a laptop computer to start is evident.

Brief Summary Text (8):

In the context of a wirelessly connected HCD, the following advantageous uses come to mind: access to e-mail, access to the Internet, access to calendars and schedules, and collaboration with co-workers. Unfortunately, most HCDs were

originally designed to function as personal computer companions or standalone data banks. By shifting the scenario to focus on direct network connectivity, these devices lose the level of processing functionality they originally had when the personal computer provided their interface to the network. Historically there have been to be two approaches to solving the problem of remote data access: (1) client side processing where the user device (a "fat" client) functions as a small computer; and (2) thin clients that operate in conjunction with server side processing.

Brief Summary Text (10):

Three variables that determine practicality to the end user are portability, affordability, and value. Fat client devices, while benefiting from additional functionality, usually suffer a decrease in portability, affordability, product practicality, and mainstream adoption. In addition, a closer look at the functionality actually being delivered by such fat client devices reveals further limitations. For example, although such devices can usually access simple POP3 and IMAP4 email accounts, they may not be sophisticated enough to negotiate corporate firewalls or communicate with proprietary servers (e.g., MICROSOFT EXCHANGE.TM. and LOTUS DOMINO.TM.) to access email or PIM data. As a result, corporate end users must maintain separate email accounts for their wireless HCDs and will have no access to corporate server-based PIM data.

Brief Summary Text (12):

With the wide-ranging proliferation of the Internet, so-called "web-based applications" have become highly prevalent. Popular sites (some examples may be HOTMAIL.TM., YAHOO! MAIL.TM., YAHOO! CALENDAR.TM., and MICROSOFT INVESTOR.TM.) provide users with a web interface to the kinds of applications that were previously only available as client side software. At one level, the term "application" seems accurate, but the usage model of a classic client-side application and a web-based application differ considerably. In contrast to the client-side model, web-based applications are stateless and non-interactive. For example, every click of the end user's mouse, selection on a menu, or update requires a reconnection to the server and a refresh of the web page. Even over the fastest Internet connections the user experience on a web-based application is arduous when compared to the persistent, interactive nature of client-side applications. Another drawback of this approach is that web-based email applications require their users to manage yet another email address. These approaches cannot function in the true sense of a desktop application, i.e., as a tool to reach individual source data instead of a service.

Brief Summary Text (13):

Some existing solution providers offer a web-based system that allows users to access their corporate data via the Internet. However, these providers require that the corporation set up a virtual private network (VPN) between the corporation's data center and the provider's service center. This may seem like a plausible enterprise solution, but the individual end user is still left without a viable alternative to traveling with a laptop computer. Furthermore, many enterprise information systems (IS) professionals are slow to adopt new technology before the functionality and demand has been generated by the people they support. End user demand will not be generated unless the specific scenario has been addressed, thus resulting in a self-perpetuating cycle.

Brief Summary Text (14):

As the Internet started gaining momentum and the static and stateless nature of web pages became apparent, new technologies such as JAVA.TM., ACTIVEX.TM., and dynamic hypertext markup language (DHTML) were developed. The growing popularity of wireless HCDs and the inadequacies of the static web view will again prompt competition related to the next development platform in the wireless market.

Description Paragraph (31):

The techniques of the present invention are preferably carried out in the context of a network data communication system. Accordingly, FIG. 1 is a schematic representation of a distributed user interface (UI) system 100 in which the techniques of the present invention may be implemented. System 100 is suitably configured to deliver information, data, control commands, and the like, from at least one server device (or system) to any number of remote end user client devices. System 100 is depicted in a generalized manner to reflect its flexible nature and ability to cooperate with any number of different communication systems, service providers, and end user devices. Although this description focuses on the processing and presentation of email data, PIM data, and "office management" data such as calendars, notes, tasks, and contact lists, the techniques of the present invention are not so limited. Indeed, the concepts described herein may be equivalently applied to the processing, delivery, and/or presentation of any suitable data format, including, but not limited to, still images, plain text, styled typography, word processor documents, spreadsheets, digital media, or any other type of information that can be transmitted via a data communication network.

Description Paragraph (32):

System 100 may include any number of client presentation devices 102, 104, 106 that communicate with at least one UI server 108. In a typical deployment, UI server 108 is implemented in a desktop or other personal computer system. In such a deployment, an individual end user maintains the UI server 108 and each of the client devices 102, 104, 106. Alternatively, UI server 108 can be implemented as any number of scalable components in a larger enterprise network environment. In this respect, a scalable enterprise solution may be configured to execute a number of network-based end user applications while concurrently supporting any number of different end users and any number of different client device platforms. In yet another deployment, a single end user with a single client device may communicate with a plurality of different UI servers representing different services, applications, or the like. For example, one client device may be supported by a desktop UI server, a UI server maintained by a service provider, a UI server maintained by an entertainment service, and the like. For the sake of simplicity and brevity, only a desktop UI server 108 is described in detail below. However, because the features and concepts of a desktop server can be equivalently applied in the context of a scalable or network-based server, the actual number of server hardware devices utilized in the system 100 may vary depending upon the particular requirements and/or specifications of the system.

Description Paragraph (33):

As used herein, a "client device" or a "presentation device" is any device or combination of devices capable of providing information to an end user of distributed UI system 100. For example, a client device 102, 104, 106 may be a personal computer, a television monitor, an Internet-ready console, a wireless telephone, a personal digital assistant (PDA), a home appliance, a component in an automobile, a video game console, or the like. The client devices may be configured in accordance with any number of conventional platforms, while using various known operating systems (OSs). For example, the client device could be a HANDSPRING VISOR.TM. running the Palm OS.TM., a POCKET PC.TM. running the WINDOWS CE OS.TM., a laptop computer running the WINDOWS 2000 OS.TM., a smartphone running a custom OEM-supplied OS, or a specialized data device built with a commercially available RTOS such as Wind River's pSOS. In practice, system 100 is particularly suited for use with wireless client devices, since it can handle the bandwidth limitations and inconsistent connections inherent in current wide-area wireless networks much better than existing alternatives. FIG. 1 depicts client device 104 as a wireless device or system.

Description Paragraph (34):

In accordance with the preferred embodiment, the client devices communicate with UI server 108 via a network 110, e.g., a local area network (LAN) a wide area network

(WAN), the Internet, or the like. Although not shown in FIG. 1, network 110 may include any number of cooperating wireless and/or wired network elements, e.g., switches, routers, hubs, wireless base stations, gateways, and the like. It should be appreciated that the present invention need not utilize network 110, e.g., any number of client devices can be connected (directly or wirelessly) to UI server 108. In the preferred embodiment, network 110 is the Internet and each of the individual client devices is configured to establish connectivity with the Internet using conventional application programs and conventional data communication protocols. For example, each client device may be configured to connect to the Internet via an internet service provider (ISP) (not shown in FIG. 1).

Description Paragraph (35):

In a practical embodiment, client devices 102, 104, 106 and UI server 108 are connected to network 110 through various communication links 112, 114. As used herein, a "communication link" may refer to the medium or channel of communication, in addition to the protocol used to carry out communication over the link. In general, a communication link may include, but is not limited to, a telephone line, a modem connection, an Internet connection, an Integrated Services Digital Network (ISDN) connection, an Asynchronous Transfer Mode (ATM) connection, a frame relay connection, an Ethernet connection, a Gigabit Ethernet connection, a Fibre Channel connection, a coaxial connection, a fiber optic connection, satellite connections (e.g., Digital Satellite Services), wireless connections, radio frequency (RF) connections, electromagnetic links, two-way paging connections, and combinations thereof.

Description Paragraph (37):

The UI server 108 preferably includes and/or communicates with one or more data sources or data servers 116, which may be configured in accordance with conventional techniques. As used herein, the data server 116 manages source data items that can be delivered to the user of the client devices. In a practical distributed UI system 100, data server 116 may manage the delivery of email, documents, PIM data, and/or any other type of data to and from the client devices. For example, the data server 116 may be realized as local, personal storage such as a MICROSOFT OUTLOOK.TM. ".pst" file on the same computer as UI server 108, or as a MICROSOFT EXCHANGE SERVER.TM., a LOTUS DOMINO SERVER.TM., a POP3 server, an IMAP server, or the like. A given data server 116 may be integral to UI server 108, it may be a distinct component maintained at the service site associated with UI server 108, or it may be maintained by a third party unrelated to the entity responsible for maintaining UI server 108. Accordingly, data server 116 may be configured to communicate with UI server 108 over a direct communication link 118 and/or via network 110 using an indirect communication link 120.

Description Paragraph (38):

A "server" is often defined as a computing device or system configured to perform any number of functions and operations associated with the management, processing, retrieval, and/or delivery of data, particularly in a network environment. Alternatively, a "server" may refer to software that performs such processes, methods, and/or techniques. As used herein, "UI server" generally refers to a computing architecture that processes data and defines display formats for the client-side UI, while executing a number of server-based applications accessed by the client devices. As in most commercially available general purpose servers, a practical UI server may be configured to run on any suitable operating system such as UNIX.TM., LINUX.TM., the APPLE MACINTOSH OS.TM., or any variant of MICROSOFT WINDOWS.TM., and it may employ any number of microprocessor devices, e.g., the PENTIUM.TM. family of processors by INTEL.TM. or the processor devices commercially available from ADVANCED MICRO DEVICES.TM., IBM.TM., SUN MICROSYSTEMS.TM., or MOTOROLA.TM..

Description Paragraph (40):

When implemented in software, various elements of the present invention (which may

reside at the client devices or at the UI server 108) are essentially the code segments that perform the various tasks. The program or code segments can be stored in a processor-readable medium or transmitted by a computer data signal embodied in a carrier wave over a transmission medium or communication path. The "processor-readable medium" or "machine-readable medium" may include any medium that can store or transfer information. Examples of the processor-readable medium include an electronic circuit, a semiconductor memory device, a ROM, a flash memory, an erasable ROM (EROM), a floppy diskette, a CD-ROM, an optical disk, a hard disk, a fiber optic medium, a radio frequency (RF) link, or the like. The computer data signal may include any signal that can propagate over a transmission medium such as electronic network channels, optical fibers, air, electromagnetic paths, or RF links. The code segments may be downloaded via computer networks such as the Internet, an intranet, a LAN, or the like.

Description Paragraph (48):

FIG. 5 is an illustration of a text edit control 500 associated with UI 300 shown in FIG. 3. (FIG. 5 also contains various label controls (From, To, and Subject) and "invisible" text edit controls associated with the labeled fields; these controls are located in the "header" area above the text edit control 500). The text edit control 500 may be generated and/or manipulated while the end user composes a new email or views a received email. The text edit control 500 may utilize multi-line edit (MLE) features to accommodate text wrapping. In practice, the text edit control may only display a portion of a text message while other portions may reside in the client cache memory or at the UI server. If end user manipulation requires the display of additional text, additional portions of the text message may be retrieved from the client cache or requested from the UI server. Upon completion of a new email message, the contents of the text edit control 500 are saved and processed for subsequent transmission to the UI server (described below).

Description Paragraph (49):

As mentioned above, the individual button controls, menu controls, and tree view control also contribute to the overall appearance of UI 300, which is rendered by the respective client device. Of course, depending upon the size and device capabilities of the client device, the particular UI may be simpler and easier to render on small displays. Briefly, the first time a connection is made from a given client device to the UI server, the display information (which may be only a few bytes) is transmitted to the client device and cached as a form definition. From then on, the UI is generated based upon the form definition. Importantly, although the controls are arranged in the layout, the form definition need not include labels or icons. For example, FIG. 6 is an illustration of an incomplete or "skeletal" UI 600 associated with an email application. Although the user will not experience the skeletal UI 600 during normal operation, the client device preferably distinguishes different UI components by keeping them in separate memory locations. This allows individual elements to be updated separately, minimizing data transfer in the event of changes at the server side. Once the various controls have been positioned to form the UI 600, the icons, labels, and menu items can be integrated from a separate cache, resulting in an intermediate UI that only lacks the actual source data items (e.g., the message list contents and the text edit fields).

Description Paragraph (89):

FIG. 8 is a schematic representation of a client icon and control data cache structure 800 that may be used by the distributed UI system. In addition to the UI controls being separated from the application, the icons, menu items, and labels that map to the controls in a form definition are also kept in another cache structure. This is done for two reasons. First, the application service provider (ASP) or wireless carrier may choose to change the look and feel of an application, or change a single item without changing the UI layout of the application. Separating the individual characteristics of a UI gives the ASP more flexibility.

The second reason is that certain icons (e.g., formatting icons or menu items) are repeated across various applications. Referencing them from a separate cache reduces the need for redundancy and maintains lower resource requirement.

Description Paragraph (107):

FIG. 9 is a flow chart of a distributed UI process 900 that may be performed by a distributed UI system as described herein. Process 900 begins when a client device establishes a connection with a UI server (task 902). The respective client and server communication interface elements may function to establish this connection (in the preferred embodiment, the connection is established over a network such as the Internet). Once the connection is made, process 900 determines whether the session corresponds to the first connection between the particular client device and the UI server (query task 904). The UI server may make this determination by, e.g., comparing a received client device identifier to a table of known or previously-connected client devices. If the current session represents the initial connection between the client device and the UI server, then an initialization process 906 (described in more detail below) is prompted. On the other hand, if the current session is a reconnection following an offline period, then a synchronization process 908 (described in more detail below) is prompted.

Description Paragraph (154):

If the command represents a client action command (query task 1808), then the command may be sent to the client UI module for further processing (task 1810). In this context, a client action command can be related to the current server-based application. The UI server can generate a client action command when necessary to have the client device perform a particular action, e.g., to display a given UI form, move or modify the attributes of a UI control, or clear the contents of a control. The client device (via, e.g., the UI module) executes the client action command and updates the UI if necessary to reflect any changes that result from the execution of the client action command.

Description Paragraph (166):

Usually, the manipulation of a UI display control will result in the display of additional data items. In other words, the current UI form will likely need to be populated with more data items. Accordingly, the client device initiates the retrieval of data items for display in the current UI form by making an appropriate request (task 2104). The client device may employ a "look ahead" technique that requests additional data from the UI server before the client device actually needs the data. For example, process 2100 may test whether a data request threshold has been exceeded (query task 2106). If this threshold has not been exceeded, then the client device may interrogate its cache to determine whether the requested data items are saved locally in the client cache (query task 2108). If the requested data items are present in the client cache, then the UI module can retrieve the data items locally from the cache and display them in the UI form (task 2110). However, if the necessary data items are not cached, then the client device will request them from the UI server.

Description Paragraph (170):

FIG. 23 is a flow chart of a process 2300 for handling the manipulation of an action control at the client device. In this context, an action control is a UI control manipulated by the user that results in the application performing an action, as opposed to updating the data displayed in the control. Typical action controls include menus and buttons, but also include data-displaying controls that have been "activated" to perform some duty, such as a double-click on an entry in a listview. Action controls result in actions such as the deletion of data items, the sending of data items, the switching of applications, or the closing of UI forms. In a practical deployment, action controls can be associated with particular UI function buttons, e.g., a "Delete" button, a "Send Message" button, or the like.

Description Paragraph (172):

If the current entry represents a "send data" command (query task 2306), then the user-entered data from the enumerated UI control(s) is formatted for delivery and placed into the client "send" queue (task 2308). Thereafter, process flow may proceed to a query task 2328 such that the next command entry can be processed. In time, the user-entered data is sent by the client send element to the UI server as described in more detail herein.

Description Paragraph (175):

If the current command represents a "delete item" command (query task 2320), then the client device updates the UI in an appropriate manner. The end user can originate a "delete item" command at different points within a UI form, e.g., from a listview control, from a message view, or from a folder tree view. As described in more detail above, the client cache may be modified if the deleted item was originally saved in the cache. In response to a "delete item" command, the client device may remove the identified or selected item from the respective control, e.g., a list control (task 2322). In addition, the deleted item and/or a suitable identifier for that item is formatted for delivery and placed into the client "send" queue (task 2324). In time, the deleted item (and/or its identifier) is sent to the UI server, which preferably updates its shadow cache to accurately reflect the current status of the client cache. Following task 2324, process flow leads to query task 2328.

Description Paragraph (179):

A client device 2402 communicates with a UI server 2404 via a suitable network 2406 such as the Internet. The client device 2402 includes a display element 2408 and a user entry element 2410 (e.g., a pointing device such as a mouse or a trackball, a keyboard, a keypad, a touchscreen, or the like). In operation, the client device 2402 renders a user interface 2412 on display element 2408. The user interface 2412 can be manipulated by the end user via user entry element 2410. For example, the end user can establish a connection with the UI server 2404, enter login data, launch and terminate server-based applications, switch between server-based applications, manipulate action controls rendered on the user interface 2412, manipulate display controls rendered on the user interface 2412, enter and edit data items associated with the user interface 2412, and perform other operations via the user interface 2412.

Current US Original Classification (1):

715/762

Current US Cross Reference Classification (1):

715/744

Current US Cross Reference Classification (2):

715/752

Current US Cross Reference Classification (3):

715/763

Current US Cross Reference Classification (4):

715/764

Current US Cross Reference Classification (5):

715/765

[Previous Doc](#)

[Next Doc](#)

[Go to Doc#](#)